originator of a communication, to encrypt the message and by the recipient of the communication to decrypt the message. It may also be used by the recipient to authenticate a message by having the sender use the secret key to compute some function such as a Message Authentication Code (MAC) based upon the message; the recipient thus can be assured of the identity of the originator, because only the sender and the recipient know the secret key used to compute the MAC. DES is an example of a symmetric cryptosystem.

[0029] In asymmetric (public key) cryptography different keys are used to encrypt and decrypt a message. Each user is associated with a pair of keys. One key (the public key) is publicly known and is used to encrypt messages destined for that user, and the other key (the private key) is known only to that user and is used to decrypt incoming messages. Since the public key need not be kept secret, it is no longer necessary to secretly convey a shared encryption key between communicating parties prior to exchanging confidential traffic or authenticating messages. RSA is the most well known asymmetric algorithm.

[0030] A digital signature, however, is a block of data appended to a message data unit, and allows the recipient to prove the origin of the message data unit and to protect it against forgery. Some asymmetric algorithms, e.g., RSA, can also provide authentication and non-repudiation through use of digital signatures. In order to sign data, the sender encrypts the data under his own private key. In order to validate the data, the recipient decrypts it with the sender's public key. If the message is successfully decrypted using the sender's public key, the message must originally have been encrypted by the sender, because the sender is the only entity that knows the corresponding private key. Using this method of signing documents, the encrypted message is bound to the signature, because the recipient cannot verify the message without decrypting the signature data block. The signature-encrypted message can then be encrypted to the recipient using the recipient's public key, as usual.

[0031] Digital signatures may also be formed using asymmetric encryption. To sign a message, the message is first digested (hashed) into a single block using a one-way hash function. Briefly, a typical one-way hash function, denoted H(M), operates on an arbitrary-length block of text or message M. The one-way hash function returns a fixed-length hash value, h, such that h=H(M), were h is of length m. One-way hash functions have special characteristics that make them one-way. Given M, for example, it is easy to compute h. Given h, it is hard to reverse the hashing process and to compute M such that H(M)=h. Further, it is very difficult to find another message, M', such that H(M)=H(M'). In essence, a one-way hash function has the property that, given the digest, it is computationally extremely difficult to construct any message that hashes to that value or to find two messages that hash to the same digest. The digest is then encrypted with the user's private key, and the result is appended to the encrypted or unencrypted message as its signature. The recipient uses the sender's public key to decrypt the signature into the hash digest. The recipient also digests (hashes) the message, which has been received either unencrypted or encrypted and then decrypted by the recipient, into a block using the same one-way hash function used by the sender. The recipient then verifies the sender's signature by checking that the decrypted hash digest is the same as the hashed message digest.

[0032] Separating the signature from the message in this way, i.e., not requiring the sender and recipient to encrypt and decrypt the entire message in order to verify the signature, greatly reduces the amount of data to be encrypted. This can be advantageous because public key algorithms are generally substantially slower than conventional algorithms, and processing the entire message in order to verify a signature requires a significant amount of time. The signature process also introduces redundancy into the message, which, because the message must hash to the specified digest, allows the recipient to detect unauthorized changes to the message.

[0033] A digital signature provides the security services of (a) integrity, because any modification of the data being signed will result in a different digest and thus a different signature; (b) origin authentication, because only the holder of the private key corresponding to the public key used for validation of the signature could have signed the message; and (c) non-repudiation, as irrevocable proof to a third party that only the signer, and not the recipient or its employees, could have created the signature. A symmetric secret key authenticator does not provide these services, since either of the two parties can create the authenticator using their shared key. The digital signatures can thereafter be used in a cryptographic system for enforcing security policies and authorization requirements in a manner that reduces risks to the users.

[0034] The biometrics processor **114**, the CPU **118**, and the cryptosystem processor **138**, may be any suitable central processing unit for executing commands and controlling the smart card **100**. The RAM portion of the RAM/ROM **122** serves as storage for calculated results and as stack memory. The ROM portion of the RAM/ROM **122** stores the operating system, fixed data, standard routines, and look up tables. Non-volatile memory (such as EPROM or EEPROM), e.g., EEPROM **134**, serves to store information that must not be lost when the card is disconnected from a power source but that must also be alterable to accommodate data specific to individual cards or any changes possible over the card lifetime. This information can include the private key **224** and biometric measurement templates and can include the public key **220**, a card serial number, a personal identification number, biometric standards or limits, authorization limits, etc. The encryption module **208** is used for performing a variety of encryption algorithms. The random number generator **204** is used to generate random keys used in the encryption algorithms. The biometrics interface **110 (FIG. 1)** is used to receive biometric data. By way of example the biometrics interface **110** may provide a contact device or an optical device to perform a finger scan or image, an iris scan or image, a retina scan or image or to determine the geometry of a hand or a face. The biometrics interface may also provide a microphone to perform a voice or speaker analysis and verification. The biometrics interface may also provide a keyboard to perform key stroke dynamic analysis or a contact to perform stroke or writing dynamic analysis and verification. The biometric data analyzer **200** is used for performing various biometric data analysis, identification and verification algorithms. The card reader interface **130** includes the software and hardware necessary for communication with the outside world. A wide